Multiple electrode (Ne1) support in the DFT+NEGF code TranSIESTA

Papior, Nick Rübner; Brandbyge, Mads

Publication date:
2015

Document Version
Publisher's PDF, also known as Version of record

Citation (APA):
Papior, N. R., & Brandbyge, M. (2015). *Multiple electrode (N_1) support in the DFT+NEGF code TranSIESTA*. Poster session presented at PSI-K 2015 Conference, San Sebastian, Spain.

Center for Nanostructured Graphene

Technical University of Denmark

DTU

Multiple electrode ($N_\mathfrak{e} \geq 1$) support in the DFT+NEGF code TranSIESTA

Nick Papior & Mads Brandbyge

DTU Dept. of Micro- and Nanotechnology & Center for Nanostructured Graphene, DK-2800 Kgs. Lyngby, Denmark

Introduction

Density functional theory based non-equilibrium Green function methods (DFT+NEGF) are by now standard for calculation of transport properties in nanostructures. In the TranSIESTA code [1, 2] the NEGF implementation is currently scaling with N^3 which limits the number of orbitals used, N being number of orbitals. In this work we present a re-implementation of TranSIESTA which scales linearly in system size (order-N) and allowing for multiple electrodes ($N_\mathfrak{e} \geq 1$) in a flexible manner [3].
In conjunction with TranSIESTA we report on an optimised tbtrans code which enables, 1) $N_\mathfrak{e} \geq 1$ electrodes, 2) interpolation of Hamiltonian between bias', 3) projection of molecular Hamiltonians, 4) custom tight-binding and 5) phonon transport.

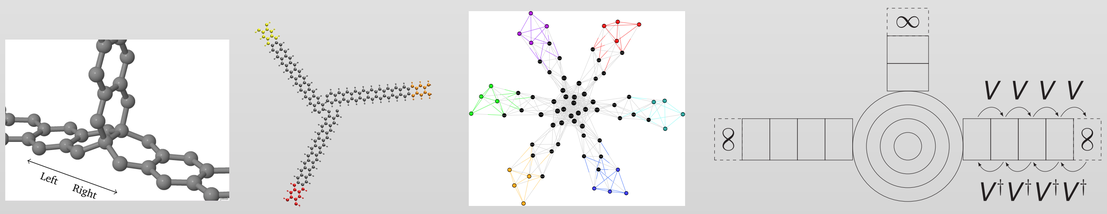
Efficiency of improved TranSIESTA DFT-NEGF method

The non-equilibrium Green function equations

$$\rho_{\text{eq}} \propto -\frac{1}{\pi}\iint_{\text{BZ}} \mathrm{d}\epsilon\mathrm{d}\mathbf{k}\, \left[\mathbf{G}_\mathbf{k}(\epsilon) - \mathbf{G}_\mathbf{k}^\dagger(\epsilon)\right] n_F(\epsilon)$$

$$\rho_{\text{neq}} \propto -\frac{1}{\pi}\iint_{\text{BZ}} \mathrm{d}\epsilon\mathrm{d}\mathbf{k} \left\{ \left[\mathbf{G}_\mathbf{k}(\epsilon) - \mathbf{G}_\mathbf{k}^\dagger(\epsilon)\right] n_{F,\mathfrak{e}_1}(\epsilon) - \mathbf{G}_\mathbf{k}(\epsilon) \sum_{\mathfrak{e}\forall \mu_\mathfrak{e}\neq\mu_{\mathfrak{e}_1}}^{\mathfrak{E}} \Gamma_{\mathfrak{e},\mathbf{k}}(\epsilon)\left[n_{F,\mathfrak{e}}(\epsilon) - n_{F,\mathfrak{e}_1}(\epsilon)\right]\mathbf{G}_\mathbf{k}^\dagger(\epsilon) \right\}$$

- Different inversion algorithms, 1) Block-tri-diagonal (BTD), 2) MUMPS, 3) LAPACK
- Efficient pivoting to obtain good scalability on BTD inversion
- Example of possible geometries



- The different available algorithms provides a very versatile NEGF code
- Hybrid parallelisation to reduce memory requirements for very large calculations

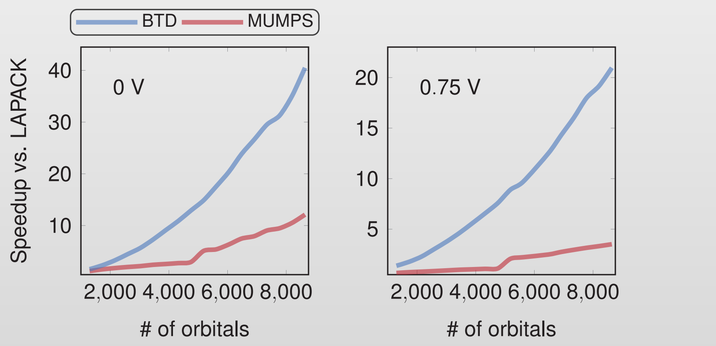


Figure: Pristine graphene calculation of differing lengths. An impressive speedup of > 40 is found for simple equilibrium calculations while > 20 for non-equilibrium calculations. The memory use is but a fraction of the full matrices which also enables much larger calculations.

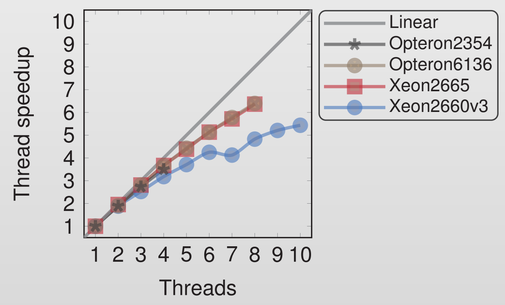
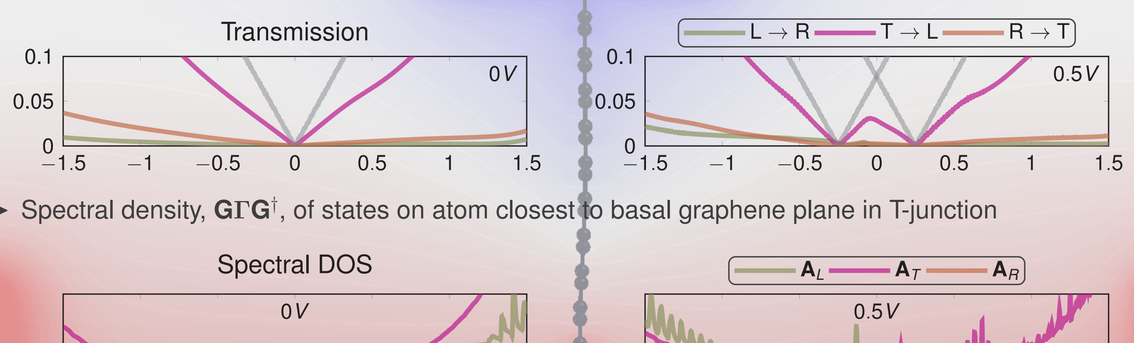


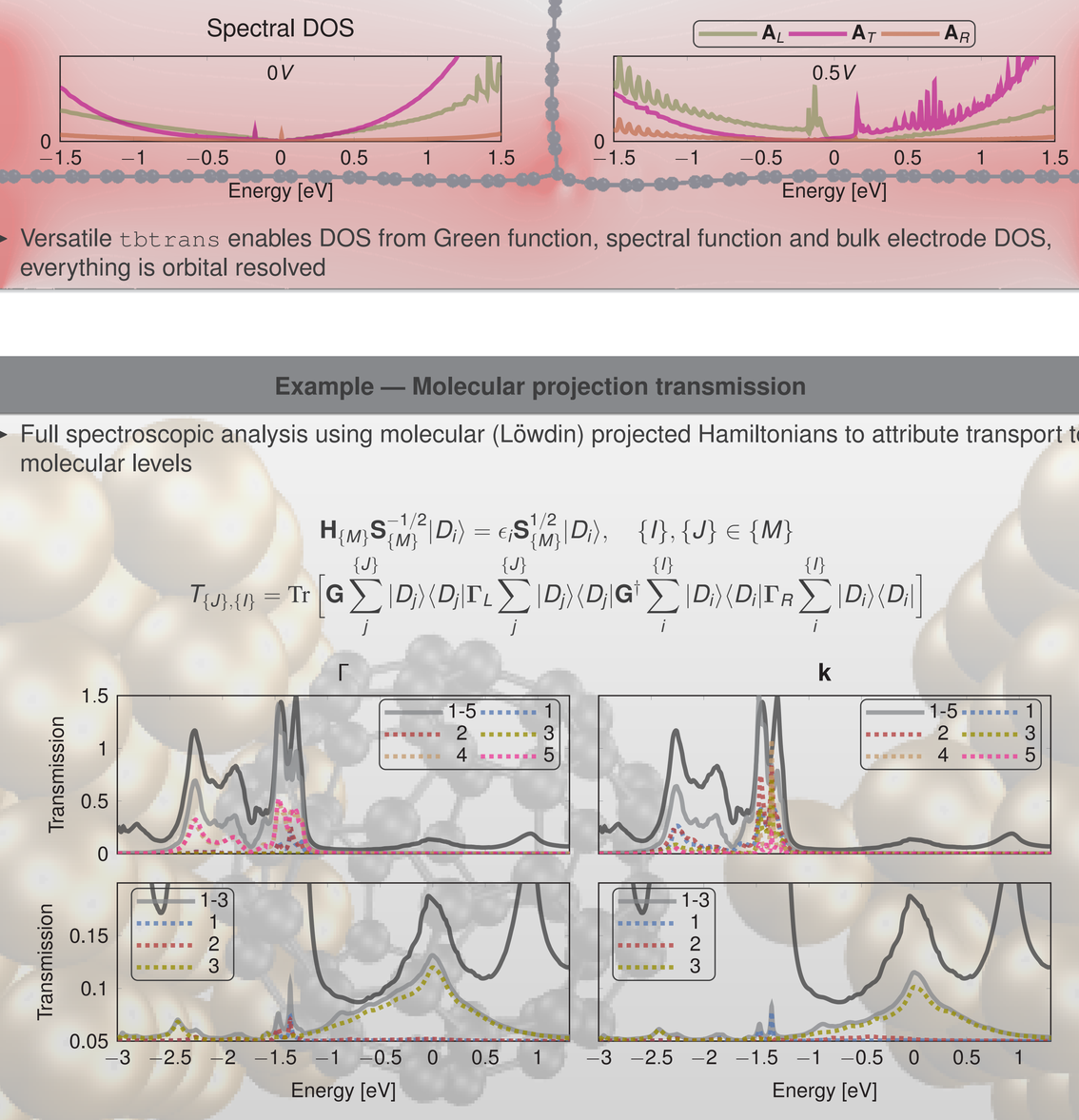
Figure: Performance gain when using threading for a very large system of 12,000 orbitals.

Example — NEGF calculation of graphene T-junction

- NEGF calculation of 3 (Left-Right-Top) electrode, periodic in the arm-chair direction.
- Compared against pristine graphene transmission spectrum



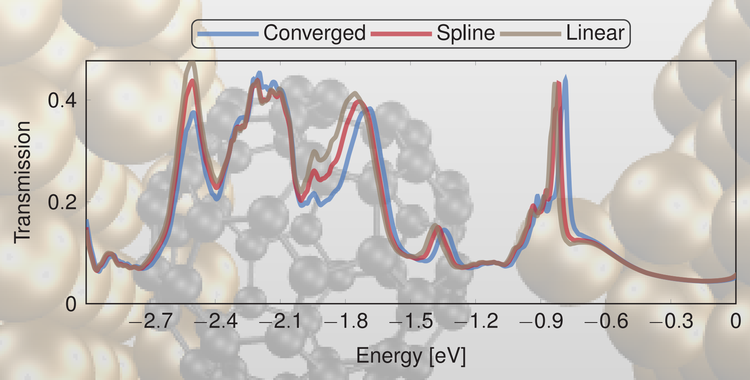
- Spectral density, $\mathbf{G}\boldsymbol{\Gamma}\mathbf{G}^\dagger$, of states on atom closest to basal graphene plane in T-junction



- Versatile tbtrans enables DOS from Green function, spectral function and bulk electrode DOS, everything is orbital resolved

Example — Interpolation of bias calculations

- Interpolation of the transmission function based on distinct bias calculations.
- Makes *I*-*V* curves feasible with a good compromise between throughput and precision.



- Interpolation using 5 NEGF calculations with steps $\{-2,-1,0,1,2\}V$ interpolated to $-1.5V$
- Spline interpolation is far superior than linear interpolation

Example — Molecular projection transmission

- Full spectroscopic analysis using molecular (Löwdin) projected Hamiltonians to attribute transport to molecular levels

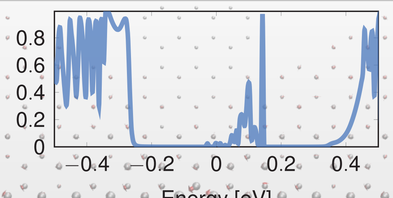
$$\mathbf{H}_{\{M\}}\mathbf{S}_{\{M\}}^{-1/2}|D_i\rangle = \epsilon_i \mathbf{S}_{\{M\}}^{1/2}|D_i\rangle, \quad \{I\},\{J\} \in \{M\}$$

$$T_{\{J\},\{I\}} = \mathrm{Tr}\left[\mathbf{G}\sum_j^{\{J\}}|D_j\rangle\langle D_j|\boldsymbol{\Gamma}_L\sum_j^{\{J\}}|D_j\rangle\langle D_j|\mathbf{G}^\dagger\sum_i^{\{I\}}|D_i\rangle\langle D_i|\boldsymbol{\Gamma}_R\sum_i^{\{I\}}|D_i\rangle\langle D_i|\right]$$

- **k** resolved projections retains better Lorentzian width ⇒ dispersion in Brillouin zone

Example — Tight-binding calculations using tbtrans

- Transmission of bow-tie graphene junction
- Flexible transport calculator, DFT *and* custom tight-binding models
- $N_\mathfrak{e}$-electrode available in highly optimised code
- Bond-currents for orthogonal basis sets




```
github.com/zerothi/sids
import sids, sids.geom as gdef, numpy as np
sq3h = 3. ** 0.5 * 0.5
alat = 1.42
C = sids.Atom(6,R=alat*sq3h*2.01)
gr = gdef.graphene(alat,C,square=True)
# Create a 100x100x4=40000 atom graphene flake
flake = gr.repeat(100,'axis=0).tile(100,'axis=1)
HS = sids.TightBinding(flake)
# Create neighbor distances, 'next-nearest
#          U   t1   t2
dR = np.array([0.1, 1.1, 2*sq3h+0.1]) * alat
for idx,idxs in flake.iter.block():
    for ia in idx:
        idx_a = flake.close(ia, dR=dR, idx=idxs)
        HS[ia,idx_a[0]] = (2.,1.)
        HS[ia,idx_a[1]] = (-1.,0.)
        HS[ia,idx_a[2]] = (-1,0.)
HS.write('DEVICE.nc')
```

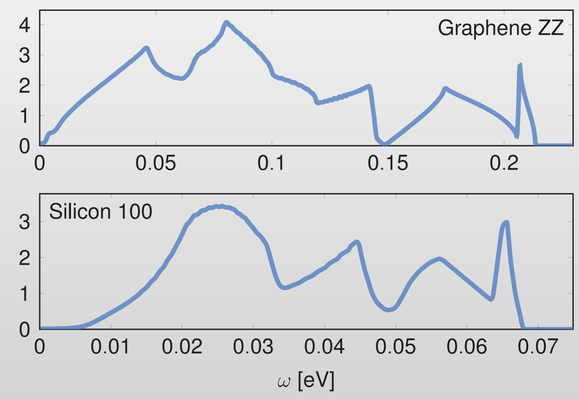
Example — Phonon transport using phtrans

- Phonon transport

$$\mathbf{G}_\mathbf{q}(\omega) = \left[\omega^2\mathbf{I} - \mathbf{D}_\mathbf{q} - \sum_\mathfrak{e}\Sigma_{\mathfrak{e},\mathbf{q}}(\omega)\right]^{-1}$$

$$\Xi_{\mathbf{q}\mathfrak{e}\mathfrak{e}'}(\omega) = \mathrm{Tr}\left[\mathbf{G}_\mathbf{q}(\omega)\boldsymbol{\Gamma}_{\mathfrak{e},\mathbf{q}}\mathbf{G}_\mathbf{q}^\dagger(\omega)\boldsymbol{\Gamma}_{\mathfrak{e}',\mathbf{q}}\right]$$

- Full $N_\mathfrak{e}$ electrode support with all capabilities of tbtrans
- Graphene phonon transport along zig-zag direction
- Bulk Silicon phonon transport along 100 direction
- Reads GULP [4] output to create Hessian (dynamical) matrix, easily extendable to other formats



Conclusion

- Full $N_\mathfrak{e} \geq 1$ NEGF calculations in TranSIESTA
- Huge performance improvement for higher throughput and larger systems
- Huge memory reduction due to implemented sparse algorithms
- Versatile transport calculator tbtrans/phtrans
  - Interpolating bias calculations
  - Molecular projected transmission spectrum
  - Implicit tight-binding calculations using simple scripting language (Python)
  - Bond-currents in orthogonal basis sets

[1] M. Brandbyge *et al. Physical Review B* 65.16 (Mar. 2002).
[2] J. M. Soler *et al. Journal of Physics: Condensed Matter* 14.11 (Mar. 2002).
[3] K. K. Saha *et al. The Journal of chemical physics* 131.16 (Oct. 2009).
[4] J. D. Gale and A. L. Rohl. *Molecular Simulation* 29 (2003).
[5] P. R. Amestoy *et al. Parallel Computing* 32.2 (Feb. 2006).
[6] P. R. Amestoy *et al. SIAM Journal on Matrix Analysis and Applications* 23.1 (Jan. 2001).
[7] E. Anderson *et al.* LAPACK Users' Guide.
[8] E. M. Godfrin. *Journal of Physics: Condensed Matter* 3.40 (Oct. 1991).
[9] E. Cuthill and J. McKee. *ACM Proceedings of the 1969 24th national conference*. 1969.
[10] M. G. Reuter and J. C. Hill. *Computational Science & Discovery* 5.1 (July 2012).
[11] W. Humphrey, A. Dalke, and K. Schulten. *Journal of molecular graphics* 14.1 (Feb. 1996).

Release in late 2015

Speak with  for more information

DTU Nanotech
Department of Micro- and Nanotechnology

nickpapior@gmail.com